# Query Completion without Query Logs for Song Search

Nitin Dua
Indian Institute of Technology
Guwahati, India 781039
+91 9954604398
nit.dua@gmail.com

Kanika Gupta Monojit Choudhury Kalika Bali
Microsoft Research Lab India
Bangalore, India 560080
+91 (80) 6658-6000
{v-kanikg, monojitc, kalikab}@microsoft.com

## ABSTRACT

We describe a new method for query completion for *Bollywood song search* without using query logs. Since song titles in non-English languages (Hindi in our case) are mostly present as Roman transliterations of the native script, both the queries and documents have a large number of valid variations. We address this problem by using a Roman to Hindi transliteration engine coupled with appropriate ranking and implementation strategies. Out of 100 test cases, our system could generate the correct suggestion for 91 queries.

## Categories and Subject Descriptors

H.3.3 [**Information Search and Retrieval**]: Query Formulation

## General Terms

Algorithms, Measurement, Experimentation

## Keywords

Query completion, Transliteration, Song search, Query log

## 1. INTRODUCTION

Song titles and lyrics are one of the most popular categories for web-search. Songs in languages using non-Roman scripts, such as Hindi, Arabic and Bengali, pose a challenge as the titles and lyrics are commonly searched and retrieved in their Roman transliterations. In many cases, no standard transliterations exist leading to a number of valid spelling variations for each word [1]. For example, *hai*, *hain*, *hey*, *he* and *hay* could refer to the same Hindi word, leading to several variations of the Hindi song title "*dil to pagal hai*". Since the other words in this title will also have quite a few commonly used Roman spellings, one ends up with a large number of possible variations for the song title. As the patterns and extent of variations in song titles are very different from usual spelling errors, IR techniques such as spelling correction, query suggestion and completion require specialized methods for these queries.

*Query completion* helps users by suggesting appropriate complete queries based on partially typed strings. Modern search engines primarily rely on *query logs* for query completion [2,3]. The techniques are based on the premise that if a user has typed a partial query *q'*, then it is very likely that he/she is trying to formulate a query q which is one of the most frequent queries in the query log beginning with the string *q'*. The basic underlying assumption, i.e., popular queries of the past will also be searched commonly in the future, is valid for song search as well. However, it may not be a good idea to assume that *q* begins with the exact string *q'* because there are so many variations in expressing the query (read song title) *q* that in the past the exact string might never have been encountered. Thus, a query log based query completion technique that is agnostic to spelling variations will require a huge amount of past queries on song titles to achieve an accuracy comparable to that for usual queries.

In this work, we describe a new query completion technique for Bollywood[1] song title search. The method does not use query logs, but only a list of song titles crawled from the Web. It uses a transliteration engine, Microsoft Indian Language Input Tool (MSILIT) [4], which generates possible Devanagari equivalents for an input string in Roman script. This allows the user to type in a Bollywood song title query in Roman script. While the user is typing, the system dynamically suggests up to 10 complete song titles that are most relevant for the partially typed query. For song search, our system outperforms the query completion feature of the most popular search engines, even though it does not have access to any query log.

## 2. METHOD

Given a partial query $q' = w_1 w_2 \ldots w_k$, where $w_1$, $w_2$ etc. are Roman transliterations of Hindi words ($w_k$ could be a partially typed word), we first generate a list of candidate song titles that the user might be searching for, compute a *relevance score* of each song title against *q'* and finally output the top 10 (or less) titles displayed in decreasing order of their relevance scores.

## 2.1 Candidate Generation

The candidate generation algorithm relies on the following two observations: (a) Even though the Roman transliterations of the titles have a large number of valid variations, in Devanagari script usually there is only one correct spelling for the titles (e.g., "दिल तो पागल है" for "*dil/dhil to/toh pagal/paagal hai/hay/he*"); (b) while searching for song titles, people usually start from the first word and type words in correct sequence.

We use the MSILIT tool to generate up to 5 possible Devanagari variations of the $w_i$'s. Let us denote these variations for $w_i$ as $d_i^1$, $d_i^2 \ldots d_i^5$. Next, we generate possible Devanagari transliterations of *q'* by combining the variations of individual words (e.g., $d_1^1 d_2^1 \ldots d_k^1$, $d_1^2 d_2^1 \ldots d_k^1$ and so on). Thus, we have at most $5^k$ combinations. Since, in practice $k$ is small (usually between 1 and 3, because if the query completion system is able to output the

[1] Bollywood is the informal term popularly used for the Hindi-language film industry based in Mumbai, India. (Wikipedia)

*WWW 2011,* March 28–April 1, 2011, Hyderabad, India.
ACM 978-1-4503-0637-9/11/03.

intended query, it does so within typing of the first 3 words), the number of combinations generated is of the order of few hundreds, and therefore, tractable. These Devanagari strings are then searched in a database of song titles in Devanagari. If no matches are returned, we search for song titles which have *all* the words of a string, but not necessarily in exact order. If even this fails to return any match, we resort to titles which have as many words of the string as possible. For each of the song titles returned by this approach, we select one of its more common Roman transliterations as a candidate for suggestion. If the above method does not return any match, we search for *q'* in a song title database in Roman script.

The databases of song titles have been built by crawling 15 popular websites for Bollywood lyrics in either Devanagari or Roman or both. The websites containing lyrics in both the scripts were used to align the song titles in the two scripts. There are about 50000 unique song titles in the database.

### 2.2 Ranking

For every candidate, we compute its *static* and *dynamic* scores. The static score tries to capture the general popularity of a song title based on the following two features: the number of crawled websites featuring this title (the higher the better), and the release date of the song (the newer the better). Some websites provide popularity rating for songs, but we found these scores to be very sparse and unreliable. Another excellent indicator of popularity is the frequency with which a song is searched for, but we do not have access to any sizeable query log in this domain to estimate this feature.

The dynamic score indicates the relevance of a candidate in the context of the partial query *q'*. If we have candidates with exact string match, this score is defined as 10 – *p*, where *p* is the positional index of the first word of the candidate which can be aligned to $w_1$ of *q'*. The dynamic scores for the cases where an exact string match is not found are computed in a similar manner. The *relevance score* is computed by adding the static and dynamic scores (computed on scales of 0-15 and 0-10).

### 2.3 Implementation Strategy

Due to repeated calls to the MSILIT engine and several database searches, generation and ranking of the candidates during runtime takes up to a few seconds. This is not desirable for a query completion system. We circumvent this problem as follows: For each unique song title, we *guess* all possible Roman variations and construct a *prefix trie* for all variations of all titles; then for every node of the trie, we run the above algorithm offline and compute the list of up to 10 most relevant song titles for the query string that leads from the root to that node. This list is stored in the trie node, so that the online computation only involves traversing the trie. The Roman variations for the Hindi words are constructed through a rule-based approach that has high recall, but low precision. MSILIT is then run on the generated strings to prune the invalid variations.

## 3. EVALUATION

We evaluated the system on 100 song title queries that were collected from 20 users who are familiar with the domain and frequently search for Bollywood songs. For each query, we generated 4 partial queries by considering up to the first 4 words and checked whether the system was able to generate the correct suggestion, i.e., the intended song title; and if so, then at what rank. The aggregate results are presented in Table 1. The last column of the table reports the number of queries (out of 100) for which the correct suggestion was not generated for any partial query up to a specific length. Users usually select the correct suggestion as soon as it is presented for the first time. Thus, the number presented in the last column provides an estimate of the number of words that a user need to type.

**Table 1. Recall statistics of the system (in %)**

| *q'* | Rank 1 | Within Rank 2 | Within Rank 3 | Within Rank 10 | Not found |
|---|---|---|---|---|---|
| $w_1$ | 18.0 | 26.0 | 34.0 | 46.0 | 54 |
| $w_1w_2$ | 57.1 | 66.2 | 69.3 | 79.6 | 17 |
| $w_1w_2w_3$ | 69.0 | 72.3 | 73.5 | 79.3 | 10 |
| $w_1w_2w_3w_4$ | 66.7 | 68.0 | 69.4 | 69.4 | 9 |

We also evaluated the query completion feature of three of the most popular commercial search engines on this data set. While our system outperforms all these search engines by a large margin for $q' = w_1$, for longer partial queries it is comparable to the best of the three engines and significantly better than the other two. These commercial search engines have access to huge query logs and a much larger part of the Web. Therefore, it is fair to assume that with access to similar resources, the current system will by far outperform the existing search engines. Of course, one should also keep in mind that we do have the unfair advantage of working on a very specific domain, that is, Bollywood song titles.

## 4. CONCLUSIONS AND FUTURE WORK

Here we described a new method for query completion for Bollywood song search. It is robust to spelling variations and does not use any query log. The technique is scalable to any sufficiently restrictive domain where the queries are transliterations of native words in a non-native script (usually Roman). Examples of similar scenarios include song, movie and book title search for languages which do not use Roman script, but Roman transliterations are fairly common (e.g., Arabic, Chinese, Japanese and most of the Indian languages). The only language dependent components of our system are the transliteration engine and rules for generating spelling variations in Roman script.

The performance of the current system can be boosted by (a) crawling more websites, (b) using query logs, (c) fine tuning the relevance score formula using machine learning techniques, and (d) improving the performance of the transliteration engine. Currently, we are working on all these aspects.

## 5. REFERENCES

[1] Sowmya V. B., Choudhury, M., Bali, K., Dasgupta, T. and Basu, A. 2010. Resource creation for training and testing of transliteration systems for Indian languages, in *Proc. of LREC'10*. pp 2902 – 2907.

[2] Meij, E., Mika, P. and Zaragoza, H. 2009. An evaluation of entity and frequency based query completion methods, in Proc of SIGIR'09. pp 678 – 679.

[3] Barouni-Ebrahimi, M. and Ghorbani, A. A. 2007. On query completion in Web search engines based on query stream mining, in *Proc of WI'07*.

[4] http://specials.msn.co.in/ilit/Hindi.aspx.